Amendments to the Claims:

1. (Currently Amended) A tangible storage medium having computer readable program code for performing data profiling, comprising [[by]]:

a first set of instructions, stored in at least one computer readable storage medium and executable by at least one processing unit, that tags[[ging]] instruction instances, that may [[accrue]]consume execution time based on loading of data, with identifiers that describe the instruction instances with source-level data object language constructs;

a second set of instructions, stored in the at least one computer readable storage medium and executable by the at least one processing unit, that identifies[[ying]] an instruction instance of the tagged instruction instances that corresponds to a runtime event detected in execution of code that includes the identified instruction instance;

a third set of instructions, stored in the at least one computer readable storage medium and executable by the at least one processing unit, that attributes[[ing]] the runtime event detected to source-level data objects describing units of data identifiable in source code, wherein the attributing is based at least in part on a predefined association between the identified instruction instance in executable code and the source-level data object language construct corresponding thereto.

2. (Previously Presented) The tangible storage medium having computer readable program code of claim 1 wherein the runtime events include sampled runtime events that statistically represent the runtime events.

3. (Previously Presented) The tangible storage medium having computer readable program code of claim 2 wherein the sampled runtime events include one or more of cache misses, cache references, data translation buffer misses, data translation buffer references, traps, and an event counter condition.

4. (Previously Presented) The tangible storage medium having computer readable program code of claim 3 wherein the event counter condition includes counter underflow or counter overflow.

5. (Previously Presented) The tangible storage medium having computer readable program code of claim 1 wherein the software tool includes a profiler, compiler, assembler, interpreter, or virtual machine.

6. (Previously Presented) The tangible storage medium having computer readable program code of claim 5 wherein the interpreter includes a byte-code interpreter.

7. (Previously Presented) The tangible storage medium having computer readable program code of claim 5 wherein the compiler includes one or more of an optimizing compiler and a byte code compiler.

8. (Previously Presented) The tangible storage medium having computer readable program code of claim 1 wherein the predefined association is included within one or more of a compiler generated code, assembler generated code, an image, and an associated separate encoding.

9. (Previously Presented) The tangible storage medium having computer readable program code of claim 1 wherein the instruction instance includes one or more of an instance of an instruction from a processor instruction set, an instance of an operation corresponding to a processor instruction set, an instance of a virtual machine instruction, or an instance of a byte code.

10. (Previously Presented) The tangible storage medium having computer readable program code of claim 1 wherein the executable code includes one or more of object code, byte code, and machine code.

11. (Previously Presented) The tangible storage medium having computer readable program code of claim 1 wherein the source-level data object language construct representation includes one or more of a class, a data type, a data size, a data type definition, a data structure, linked object,, and a member of a data structure, static variables, automatic variables, memory segment.

12. (Previously Presented) The tangible storage medium having computer readable program code of claim 1 wherein the corresponding language of the language construct includes a source-level language or an intermediate level language.

13. (Currently Amended) The tangible storage medium having computer readable program code of claim 1 further comprising<u>:</u> ~~the software tool~~

<u>a fourth set of instructions, stored in the at least one computer readable storage medium and executable by the at least one processing unit, that</u> aggregat<u>es</u>[[ing]] runtime events based on the source-level data objects.

14. (Currently Amended) The tangible storage medium having computer readable program code of claim 13 further comprising: ~~that also displays~~

a fifth set of instructions, stored in the at least one computer readable storage medium and executable by the at least one processing unit, that displays aggregated runtime events.

15. (Currently Amended) The tangible storage medium having computer readable program code of claim 1, further comprising: ~~that also aggregates~~

a fourth set of instructions, stored in the at least one computer readable storage medium and executable by the at least one processing unit, that aggregates profile data for the code based on the source-level data object language construct representation.

16. (Currently Amended) A method of data profiling code, the method comprising:

tagging instruction instances stored in at least one computer readable storage medium utilizing at least one processing unit, that may [[accrue]]consume execution time based on loading of data, with identifiers that describe the instruction instances with source-level data object language constructs;

for a sampled runtime event detected in execution of the code, identifying an instruction instance of the tagged instruction instances that corresponds to the sampled runtime event detected in execution of code that includes the identified instruction instance utilizing the at least one processing unit; and

attributing the sampled runtime event detected to source-level data objects describing units of data identifiable in source code utilizing the at least one processing unit, wherein the attributing is based at least in part on a predefined association between the identified instruction instance in executable code and the source-level data object language construct corresponding thereto.

17. (Original) The method of claim 16 wherein the predefined association is included within one or more of a compiler generated code, assembler generated code, an image, or an associated separate encoding.

18. (Original) The method of claim 16 wherein the corresponding operation instance includes one or more of an instance of an instruction from a processor instruction set, an instance of an operation corresponding to a processor instruction set, an instance of a virtual machine instruction, and an instance of a byte code.

19. (Original) The method of claim 16 wherein the code includes one or more of object code, byte code, and machine code.

20. (Original) The method of claim 16 wherein the source-level data object language construct includes one or more of a data type, a data type definition, a data structure, a data size, and a function.

21. (Original) The method of claim 16 wherein the language of the language construct includes a source-level language or an intermediate level language.

22. (Original) The method of claim 16 wherein the sampled runtime event includes one or more of a cache miss, cache reference, data translation buffer miss, data translation buffer reference, and an event counter condition.

23. (Original) The method of claim 22 wherein the counter event condition includes counter underflow or counter overflow.

24. (Previously Presented) The method of claim 16 embodied in a computer program product encoded on one or more tangible storage machine-readable media.

25. (Original) The method of claim 16 further comprising aggregating profiling data for the code based at least in part on the language construct.

26. (Currently Amended) A method of data profiling code, the method comprising:

associating a source-level data object language construct for a source-level data object by tagging an instruction instance stored in at least one computer readable storage medium that may [[accrue]]consume execution time based on loading of data utilizing at least one processing unit, with identifiers that describe the instruction instance with a source-level data object language construct; and

attributing a sampled runtime event to the source-level data object language construct describing units of data identifiable in source code utilizing the at least one processing unit, wherein the attributing is based at least in part on the association between the instruction instance in executable code and the source-level data object language construct corresponding thereto.

27. (Original) The method of claim 26 wherein the code includes one or more of object code, byte code, and machine code.

28. (Original) The method of claim 26 wherein the source-level data object language construct includes one or more of a data structure, classes, variable instances, objects, a member of a data structure, and a statically linked object.

29. (Original) The method of claim 26 wherein the instruction instance includes a load type instruction.

30. (Original) The method of claim 26 wherein the language construct includes one or more lexical tokens.

31. (Original) The method of claim 30 wherein the lexical tokens include one or more of identifiers and literals.

32. (Original) The method of claim 26 wherein the source-level data object is represented in a source-level language or an intermediate level language.

33. (Original) The method of claim 26 wherein the sampled runtime event includes one or more of a cache miss, cache reference, a data translation buffer miss, data translation buffer reference, and a counter condition event.

34. (Original) The method of claim 33 wherein cache miss includes a data cache miss, an instruction cache miss, a unified cache miss, and an external cache miss.

35. (Original) The method of claim 33 wherein cache references includes one or more of a data cache reference, an instruction cache reference, a unified cache reference, and an external cache reference.

36. (Original) The method of claim 33 wherein the counter condition event includes a counter overflow event or a counter underflow event.

37. (Currently Amended) The method of claim 26 further comprising backtracking from a second instruction instance to the instruction instance, utilizing the at least one processing unit, after detecting the sampled runtime event.

38. (Previously Presented) The method of claim 26 embodied in a computer program product encoded on one or more tangible storage machine-readable media.

39. (Currently Amended) The method of claim 26 further comprising aggregating profile data for the code based at least in part on the language construct utilizing the at least one processing unit.

40-48. (Cancelled)

49. (Currently Amended) An apparatus comprising:

a set of one or more processors;

a memory coupled with the set of processors; and

a tangible storage machine-readable media coupled with the set of processors, the tangible storage machine-readable media having stored therein a set of data profiling instructions to cause the set of processors to;

tag instruction instances, that may [[accrue]]consume execution time based on loading of data, with identifiers that describe the instruction instances with source-level data object language constructs; and

attribute a runtime event to a source-level data object language construct that corresponds to a source-level data object, wherein the source-level data object describes units of data identifiable in source code, that is associated with an instance of an instruction of the tagged instruction instances, which corresponds to the runtime event.

50. (Original) The apparatus of claim 49 wherein at least one of the set of processors includes a data cache.

51. (Original) The apparatus of claim 49 wherein the machine-readable media includes propagated signal, optical storage, or magnetic storage.

52. (Original) The apparatus of claim 49 wherein the set of data profiling instructions further cause, when executed, the set of processors to aggregate profile data for the code based on the source-level data object language construct.